

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-062580

(43)Date of publication of application : 07.03.1997

(51)Int.Cl.

G06F 12/08
G06F 15/163

(21)Application number : 07-221638

(71)Applicant : CANON INC

(22)Date of filing : 30.08.1995

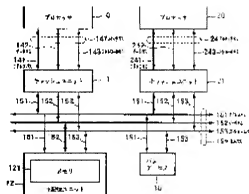
(72)Inventor : NAKAMURA SHUICHI
FUKUI TOSHIYUKI
HAMAGUCHI KAZUMASA

(54) MULTI-PROCESSOR DEVICE

(57)Abstract:

PROBLEM TO BE SOLVED: To avoid the degradation in use efficiency of a cross coupling network by maintaining the consistency of data blocks existing in each cache memory based on issue of a prescribed instruction at the time of issue of this instruction from the processing of each processor.

SOLUTION: Cache units 11 and 21 are connected to a main storage unit 12 and a bus arbiter 16. Cache units 11 and 21 update data entries of data blocks in units themselves or reflect them on the main storage unit 12 based on requests of processors and snoop the address information or the like flowing on a local bus 15 to execute the operation of cache consistency maintenance or the like. The bus arbiter 16 arbitrates the use right of the local bus 15. When the processing of each of processors 10 and 20 issues the prescribed instruction, the consistency of data blocks existing in each cache memory is maintained based on this instruction issue.



(51) Int. Cl. ⁴	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 12/08	3 1 0	7623-5B	G 0 6 F 12/08	3 1 0 B
15/163			15/16	3 2 0 K

審査請求 未請求 請求項の数 4 O L (全 11 頁)

(21) 出願番号 特願平7-221638

(22) 出願日 平成7年(1995)8月30日

(71) 出願人 000001007

キヤノン株式会社
東京都大田区下丸子3丁目30番2号

(72) 発明者 中村 秀一

東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

(72) 発明者 福井 俊之

東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

(72) 発明者 濱口 一正

東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

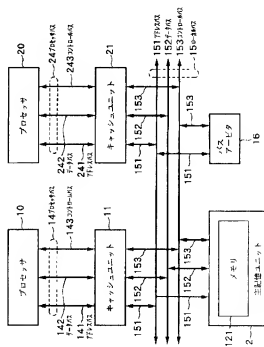
(74) 代理人 弁理士 大塚 康徳 (外1名)

(54) 【発明の名称】 マルチプロセッサ装置

(57) 【要約】

【課題】 キャッシュの一貫性保持動作の実施を都合の良い時点まで遅延させ、共有データブロックに対してのみ一貫性保持を行うことで、例えば、緩いメモリ・コンシステンシ・モデルのシステムに、同期ポイント時点における相互結合網上のトラフィックの集中による相互結合網の利用効率の低下を避け、より性能の高いマルチプロセッサ装置を提供する。

【解決手段】 プロセッサ10が例えばSTORE命令を行った際には、キャッシュユニット11にデータが格納されるが、この時点では主記憶ユニット12にはライトバックせじ、プロセッサ10が特別な同期命令を発行した場合に、はじめてライトバックする。



【特許請求の範囲】

【請求項1】 複数のプロセッサ及びそれらに付随するキャッシュメモリ、主記憶装置、及び、キャッシュ・メモリと記憶装置間を相互に接続する結合網からなる並列計算機システムにおいて、

各プロセッサが所定の命令発行を行った場合に、当該命令発行に基づいて各キャッシュ・メモリ中に存在するデータブロックの一貫性保持を行うことを特徴とするマルチプロセッサ装置。

【請求項2】 前記一貫性保持は、

各プロセッサから発行されたメモリ・トランザクションが、他のキャッシュ中に共有される可能性があるデータブロックに対するメモリ・トランザクションであることを識別する識別手段と、

前記共有データブロックとして識別されたメモリ・トランザクションによって格納されたキャッシュ中のデータブロックが共有データブロックであることを保持する保持手段と、

前記保持手段により保持されている情報を用いて一貫性保持動作を実施するデータブロックを選択する選択手段を備えることにより、上記プロセッサが予め定めた段階において、共有データブロックに対してのみシステムにおける一貫性保持動作を実施することを特徴とする請求項1記載のマルチプロセッサ装置。

【請求項3】 上記システムは緩いメモリ・コンシステンシーモデルをとることを特徴とする請求項1及び2に記載のマルチプロセッサ装置。

【請求項4】 前記共有データブロックは、クリティカルセクション内で読み書きされるデータブロックであることを特徴とする請求項第2項に記載のマルチプロセッサ装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明はマルチプロセッサ装置、詳しくは、各プロセッサがそれぞれキャッシュを備えたマルチプロセッサ装置に関するものである。

【0002】

【従来の技術】並列計算機システムにおいては、プロセッサから発行される主記憶に対するアクセス要求に高速に応じるため、及び相互結合網のトラフィックを減じるために、各プロセッサにキャッシュ・メモリを付随させることが多い。各プロセッサから発行されるメモリ・アクセスはキャッシュ・メモリ（及びキャッシュメモリコントローラ）を介して行われ、キャッシュ・メモリ中にはそれぞれメモリ・アクセス対象のデータ・ブロックのコピーが置かれることになる。

【0003】並列計算機システムにおいては、複数のキャッシュ・メモリ中に同一データ・ブロックのコピーが各々存在する状況が生じ得るが、それらコピー間の一貫性を保証するために、従来様々な方法が考案・実現さ

れている。

【0004】プロセッサ間やプロセッサ・主記憶間を相互に接続する結合網に、全てのトランザクションが監視可能であるバスのようなものを用いた並列計算機システムにおいては、スヌープ方式が一般的である。スヌープ方式は、キャッシュ・メモリが結合網上に発行される全トランザクションを監視し、トランザクション対象のデータ・ブロックのコピーが自キャッシュ・メモリ中に存在していた場合は、必要な一貫性保持動作を施すものである。

【0005】また、プロセッサ間やプロセッサ・主記憶間を相互に接続する結合網に、全てのトランザクションを監視することが困難である並列計算機システムにおいては、ディレクトリ方式が用いられる。ディレクトリ方式は、データ・ブロック単位、あるいはそれに相当する単位毎に、いずれのキャッシュ・メモリ中にそのコピーが存在するかというキャッシング情報を、ディレクトリと呼ばれる記憶装置に格納・管理しておき、プロセッサからのトランザクション発行時にはディレクトリから得られるキャッシング情報をもとにして、トランザクション対象データ・ブロックのコピーを有するキャッシュ・メモリにトランザクションの発生を通知し、コピー間の一貫性保持を図るものである。

【0006】

【発明が解決しようとする課題】従来、並列計算機システムにおける複数のキャッシュ・メモリ中に存在するコピー間の一貫性をとるための動作は、上述の通りトランザクション毎に行われるものであった。しかし、これは、メモリに対するアクセス・レイテンシを抑えるために様々な考案/実現されている緩いメモリ・コンシステンシーモデルにはそぐわないものである。

【0007】一般に、緩いメモリ・コンシステンシーモデルでは、処理の過程に同期ポイントを定め、処理が同期ポイントに達した時点で、それまでに発行したメモリ・トランザクションをシステム中に反映させることを義務付けている。このことは、同期ポイント以前には各メモリ・トランザクション結果を反映させる必要がないことを意味する。即ち、緩いメモリ・コンシステンシーモデルを採る並列計算機システムにおいて従来のキャッシュ一貫性保持手法を用いた場合、その時点では不要な一貫性保持動作がトランザクション毎に入ることとなり、そのオーバーヘッドは、緩いメモリ・コンシステンシーモデルの目的に反し、不用意にメモリ・アクセス・レイテンシを高めていると言える。

【0008】しかし、キャッシュの一貫性保持動作の実施を、緩いメモリ・コンシステンシーモデルでメモリ・トランザクションを反映させる必要が生じる同期ポイントの時点にまで遅延させることによって、不要なキャッシュ一貫性保持動作によるオーバーヘッドを削減するようなシステムにおいては、同期ポイントの時点において集

中的にキャッシュの一貫性保持動作が行われるため、同期ポイントの時点で相互結合網上に集中的にトラフィックが発生することになり、その結果、相互結合網の利用効率が極端に低下する恐れがあった。

【0009】

【課題を解決するための手段】本発明はかかる問題点に鑑みなされたものであり、キャッシュの一貫性保持動作の実施を都合の良い時点まで遅延させることで、例えば、緩いメモリ・コンシステンシ・モデルのシステムに、同期ポイント時点における相互結合網上のトラフィックの集中による相互結合網の利用効率の低下を避け、より性能の高いマルチプロセッサ装置を提供しようとするものである。

【0010】この課題を解決するため、例えば本発明は以下の構成を備える。すなわち、複数のプロセッサ及びそれらに付随するキャッシュメモリ、主記憶装置、及び、キャッシュ・メモリと記憶装置間を相互に接続する結合網からなる並列計算機システムにおいて、各プロセッサの処理が所定の命令発行を行った場合に、当該命令発行に基づいてキャッシュ・メモリ中に存在するデータブロックの一貫性保持を行うことを特徴とする。

【0011】ここで本発明の好適な実施形態に従えば、一貫性保持は、各プロセッサから発行されたメモリ・トラザクションが、他のキャッシュ中に共有される可能性があるデータブロックに対するメモリ・トラザクションであることを識別する識別手段と、前記共有データブロックと識別されたメモリ・トラザクションによって格納されたキャッシュ中のデータブロックが共有データブロックであることを保持する保持手段と、前記保持手段により保持されている情報を用いて一貫性保持動作を実施するデータブロックを選択する選択手段を備えることが望ましい。これによって、プロセッサが予め定めた段階において、共有データブロックに対してのみシステムにおける一貫性保持動作を実施する。

【0012】

【発明の実施の形態】以下、図面を参照して本発明の実施の形態を詳細に説明する。図1は、本発明を実現するためのシステムの第1の実施形態の構成を示すブロック図である。図中、10、20はプロセッサであり、それぞれプロセッサバス14、24を介してキャッシュユニット11、21に接続されている。

【0013】プロセッサ10は、メモリ・アクセスを多重発行することが可能であり、多重発行されたメモリ・アクセスを緩いメモリ・コンシステンシ・モデルによるデータ一貫性保証の下で完了させるための特別な命令(同期命令)を持つものとする。

【0014】また、プロセッサ10は、共有データブロック(例えば、クリティカルセクション内で読み書きされるデータブロック)に対するメモリ・アクセスを開始することを宣言する特別な命令(以下共有データブロッ

クアクセス開始命令)、共有データブロックに対するメモリ・アクセスを終了することを宣言する特別な命令(以下共有データブロックアクセス終了命令)を持ち、これらの命令はプロセッサコントロールバス143(後述)によって外部モジュールに識別されるものとするが、共有データブロック、非共有データブロックに対するメモリ・アクセスの開始、終了を明示する機構は本実施の形態に制限されるものではない。また、共有データブロックアクセス開始命令は共有データブロックに対する排他的な使用権を確保するものであり、共有データブロックアクセス終了命令は共有データブロックに対する排他的な使用権を解放するものであるとし、共有データブロックアクセス開始命令、共有データブロックアクセス終了命令は、ともに主記憶ユニットまで完了の通知が必要であるものとする。

【0015】またキャッシュユニット11、21は、それぞれローカルバス15を介して主記憶ユニット12及び、バスアービタ16と接続される。またキャッシュユニット11、21は、プロセッサの要求に基づいて、キャッシュユニット11、21の内部のデータブロックのデータエントリの更新を行ったり、主記憶ユニット12に反映させたり等と共に、ローカルバス15上を流れるアドレス情報等をスヌープしてキャッシュ一貫性保持等の動作を実施する。バスアービタ16はローカルバス15の利用権を調停するためのものである。本実施の形態では、共有バスに接続されたマルチプロセッサシステム構成を例にあげているが、これらの構成は本実施の形態に制限されるものではない。

【0016】本実施の形態では、プロセッサから明示的に与えられる同期命令発行時点でデータブロックの一貫性が保証されるような緩いメモリ・コンシステンシ・モデルを採用した一貫性制御が行われる図1に示すような構成の情報処理システムにおいて、キャッシュの一貫性保持動作を制御するキャッシュ制御シーケンスが、

(1)共有データブロックに対するメモリ・アクセスを開始する命令(共有データブロックアクセス開始命令)がプロセッサから発行された場合に、以後のメモリ・アクセスに対して共有データブロックに対するメモリ・アクセスを終了する命令(共有データブロックアクセス終了命令)がプロセッサから発行されるまで、共有データブロック・アクセスフラグをセットする。

【0017】(2)主記憶にその値を反映していない状態(以下DIRTY状態)にあるキャッシュのデータブロックを、共有データブロックであるかという判断し、同期命令発行時点で共有データブロックのみの主記憶へのライトバック処理、及び必要ならば一貫性保持動作を実行する。

【0018】という動作を行うことによって、同期命令発行時点で集中的に発生する主記憶へのライトバック処理、ライトバックや一貫性保持動作・トラフィックのうち

非共有データブロックに対するトラフィックを発生させないことを実現している。

【0019】図2は、本実施の形態の一部分であるキャッシュユニットの構成を示す図である。図2においては、キャッシュユニット11を例にあげているが、キャッシュユニット21についても同様な構成をとるものである。

【0020】図中、144はプロセッサアドレスバス141と接続するためのプロセッサアドレスバスインタフェースであり、145はプロセッサデータバス142と接続するためのプロセッサデータバスインタフェースであり、146はプロセッサコントロールバス143と接続するためのプロセッサコントロールバスインタフェースである。

【0021】また、154はローカルアドレスバス151と接続するためのローカルアドレスバスインタフェースであり、155はローカルデータバス152と接続するためのローカルデータバスインタフェースであり、156はローカルコントロールバス153と接続するためのローカルコントロールバスインタフェースである。

【0022】115はデータを保持するためのデータエントリであり、112はデータエントリ115のアドレスを保持するためのアドレスタグであり、113はデータエントリ115の状態を保持するための状態フラグであり、114はデータエントリ115が共有データブロックであるかどうかを保持するための共有データブロックフラグである。これらの部分は例えばSRAMのような記憶素子の集合であるとするが、これは本実施形態に制限されるものではない。

【0023】116はアドレスタグ112の内容とプロセッサアドレスバス141、ローカルアドレスバス151とのアドレスを比較する比較器である。また、117は比較器116の比較結果からデータエントリ内のデータを選択する選択器である。

【0024】また、図中、118はプロセッサ10が共有データブロックに対するアクセス中であることを示す共有データブロック・アクセスフラグであり、プロセッサ10から発行された共有データブロックアクセス開始命令が完了するとセットされ、プロセッサ10から発行された共有データブロックアクセス終了命令が完了するとリセットされるものとする。

【0025】111はキャッシュユニット内の各モジュールを制御するキャッシュ制御シーケンサである。

【0026】本実施の形態では、キャッシュユニットは1キャッシュ・ラインあたり256バイト、1セットあたり16キャッシュ・ラインの2ウェイ・セット・アソシアティブ構成をとっているが、これらの構成は本実施の形態に制限されるものではない。

【0027】図3は、キャッシュユニット11内のアドレスタグ112、状態フラグ113、共有データブロック

クフラグ114及びデータエントリ115の関係、及びアドレス比較時のアドレス・フィールドを示したものである。

【0028】図示において、32ビットのアドレス0x f8000701 (0xは16進数を示す)がプロセッサアドレスバス141又はローカルアドレスバス151から渡された場合、アドレス比較がどのように行われるかを説明する。

【0029】本実施の形態では、キャッシュユニットは1キャッシュ・ラインあたり256バイト、1セットあたり16キャッシュ・ラインの2ウェイ・セット・アソシアティブ構成をとっているため、アドレスタグ112は20ビット、キャッシュ・ラインの選択に使用されるインデックスは4ビット、キャッシュ・ライン内のデータブロックの位置を指定するオフセットは8ビットのフィールドを持つ。渡されたアドレスの下位8ビットはオフセットフィールドであり、キャッシュ・ライン内のデータブロックの位置を指定する。渡されたアドレスの上位20ビットはアドレスタグフィールドであり、アドレスタグ112と比較される。尚、渡されたアドレスのオフセットフィールドとアドレスタグフィールドを除いた部分はインデックスフィールドであり、セット内のキャッシュ・ラインの位置を指定する。

【0030】比較器116はインデックスフィールドにより指定されたキャッシュ・ラインのアドレスタグに格納されている値とプロセッサアドレスバス141又はローカルアドレスバス151から渡されたアドレスの上位20ビットとを比較する。上記比較結果が一致しており、かつ、インデックスフィールドにより指定されたキャッシュ・ラインの状態フラグ(状態フラグの詳細については後述)が有効状態(CLEAN、又はDIRTY)の場合、キャッシュ・ヒットとする(トランザクションがREADの場合はキャッシュ・リードヒット、WRITEの場合はキャッシュ・ライトヒットと呼ぶ)。キャッシュ・ヒット以外の場合はキャッシュ・ミスとする(トランザクションがREADの場合はキャッシュ・リードミス、WRITEの場合はキャッシュ・ライトミスと呼ぶ)。

【0031】図3において、アドレス0x f8000701のオフセットフィールドは0x01であるため、キャッシュ・ライン内のオフセットは1となる。アドレス0x f8000701のインデックスフィールドは0x7であるため、セット内のキャッシュ・ラインインデックスは7となる。アドレス0x f8000701のアドレスタグフィールドは0x f8000であり、キャッシュ・ラインインデックス7に格納されているアドレスタグと比較される。

【0032】図4は、各メモリランザクション実行時の状態フラグの状態遷移図を示したものである(以下キャッシュユニット11の状態フラグとして説明する)。図4において、状態INVALIDは当該状態フラグの管理す

るデータエントリが無効であることを示す。状態CLEANは当該状態フラグが管理するデータエントリが主記憶ユニットからLOADされた後、1度も書き換えられていないことを示す。但し、当該データエントリには主記憶ユニットと同一の値が格納されているが、他のキャッシュユニットのデータエントリには最新の値が格納されている可能性がある。状態DIRTYは当該状態フラグが管理するデータエントリが主記憶ユニットからLOADされた後、1度以上最新の値で書き換えられ、かつ、主記憶にその値を反映していないことを示す。当該データエントリには最新の値が格納されている。各状態からの遷移条件は以下のとおりであるとする。

【0033】(1) INVALID状態にあるデータブロックに対してプロセッサ10からLOAD命令が発行された場合、状態フラグはCLEANに遷移する。

【0034】(2) INVALID状態にあるデータブロックに対してプロセッサ10からSTORE命令が発行された場合、一旦キャッシュ・リードミス処理が実行され、状態フラグがCLEANに遷移した後、キャッシュ・ライトヒット処理が実行され、状態フラグはDIRTYに遷移する。

【0035】(3) CLEAN状態にあるデータブロックに対してプロセッサ10からLOAD命令が発行された場合、状態フラグはCLEANに遷移する。

【0036】(4) CLEAN状態にあるデータブロックに対してプロセッサ10からSTORE命令が発行された場合、状態フラグはDIRTYに遷移する。

【0037】(5) DIRTY状態にあるデータブロックに対して一貫性保持動作が実行された場合、状態フラグはINVALIDに遷移する。

【0038】(6) DIRTY状態にあるデータブロックに対してプロセッサ10からLOAD命令が発行された場合、状態フラグはDIRTYに遷移する。

【0039】(7) DIRTY状態にあるデータブロックに対してプロセッサ10からSTORE命令が発行された状態フラグはINVALIDに遷移する。

【0040】(8) DIRTY状態にあるデータブロックに対してプロセッサ10から同期命令が発行された場合、状態フラグはCLEANに遷移する。

【0041】(9) DIRTY状態にあるデータブロックに対して主記憶へのライトバック処理が実行された場合、状態フラグはCLEANに遷移する。

【0042】(10) DIRTY状態にあるデータブロックに対して一貫性保持動作が実行された場合、状態フラグはINVALIDに遷移する。

【0043】図5は、LOAD命令実行の際の制御手順を示したものである(以下LOAD命令がプロセッサ10から発行されたものとして説明する)。

【0044】図5において、プロセッサ10から発行されるLOAD命令に対してアドレス比較を行い(ステップS1)、キャッシュ・リードヒットした場合、キャッシュ

ユニット11は、プロセッサ10に対して自身に記憶されているデータブロックを供給する(ステップS2)。

【0045】また、プロセッサ10から発行されるLOAD命令に対してアドレス比較を行い、キャッシュ・リードミスした場合、キャッシュユニット11は、ローカルバス15に対してリード要求を発行する(ステップS3)。

この場合、キャッシュ・リードミスのデータブロックがキャッシュユニット11に供給されるまで、キャッシュユニット11はプロセッサ10に対して当該データブロックの供給を行わない。キャッシュユニット11は、ローカルアドレスバス151に対して当該リードアクセスのアドレスを転送し、ローカルコントローラバス153に対してリード要求を発行し、ローカルデータバス152にてデータブロックが供給されるまで一貫性保持動作に対するサービスを及びプロセッサからのアクセス要求に対するサービスを停止するものとするが、これは本実施の形態に制限されるものではない。主記憶ユニット12は、リード要求、及び当該リードアクセスのアドレスを受け付けてローカルデータバス152にてデータブロックを供給する(ステップS4)。キャッシュユニット11は、ローカルデータバス152に供給されたデータブロックをキャッシュユニット11内の当該データブロックのエントリに置換し、当該データブロックの状態フラグ113をCLEANとする(ステップS5)。共有データブロック・アクセスフラグ118がセットされている場合、キャッシュユニット11内の当該データブロックの共有データブロックフラグ114をセットする。共有データブロック・アクセスフラグ118がリセットされている場合、キャッシュユニット11内の当該データブロックの共有データブロックフラグ114をリセットする。キャッシュユニット11は、プロセッサ10に対してデータブロックを供給する(ステップS6)。

【0046】次に、STORE命令実行の際の制御手順を図6に従って説明する(以下STORE命令がプロセッサ10から発行されたものとして説明する)。

【0047】図6において、プロセッサ10から発行されるSTORE命令に対してアドレス比較を行い(ステップS10)、キャッシュ・ライトヒットした場合、当該データブロックの状態フラグをDIRTY状態とする(ステップS12、13)。プロセッサ10から発行されるSTORE命令に対してアドレス比較を行い、キャッシュ・ライトミスした場合、キャッシュ・リードミス処理を行った後、キャッシュ・ライトヒット処理を行うものとする(ステップS11)。

【0048】図7は、同期命令実行の際の制御手順を示したものである(以下同期命令がプロセッサ10から発行されたものとして説明する)。

【0049】図示において、キャッシュユニット11に共有データブロックフラグ114がセットされ、かつ、DIRTY状態のデータブロックが1つ以上存在し、ステッ

フS20)、かつ、プロセッサ10から同期命令が発行された場合、当該DIRTYブロックの主記憶へのライトバック処理を実行し、当該データブロックの状態フラグをCLEANにする(ステップS22、23)。

【0050】本実施形態ではDIRTYブロックの主記憶へのライトバック処理(DIRTYブロックがなくなるまで繰り返し実行されるものとするが、これは本実施形態に制限されるものではない。

【0051】図8は、主記憶へのライトバック処理実行の際の制御手順を示したものである(以下主記憶へのライトバック処理がキャッシュユニット11から発行されたものとして説明する)。

【0052】図8において、キャッシュユニット11は、ローカルコントロールバス153に対してライトバック要求元情報、及びローカルアドレスバス151に対して当該ライトバックアクセスのアドレスを転送し、ローカルコントロールバス153に対してライトバック要求を発行し、ローカルデータバス152にライトバックするデータブロックを供給する(ステップS30)。そして、ライトバックしたデータブロックが当該主記憶ユニットのエントリに供給され、必要な一貫性保持動作が完了し、主記憶へのライトバック処理が完了するまで実行を停止する。

【0053】主記憶ユニット12は、リード要求、及び当該ライトバックアクセスのアドレス及びローカルデータバス152に供給されているデータブロックを受け付けて当該データブロックのエントリにリードする。また、同時にキャッシュユニット11以外のキャッシュユニットにもデータのコピーが有効状態(CLEAN状態、もしくはDIRTY状態)で保持されている場合は、有効状態で保持しているキャッシュユニットはローカルバス151に転送されたアドレスをスヌープして有効状態で保持しているデータブロックに対して一貫性保持動作を実行する(ステップS31～ステップS34)。

【0054】本実施形態では一貫性保持動作は無効化型のトランザクションとするが、これは本実施形態に制限されるものではない。

【0055】図9は、データブロックの置換実行の際の制御手順を示したものである(以下キャッシュユニット11がデータブロックの置換を実行するものとして説明する)。

【0056】図9において、キャッシュユニット11は、データブロックの置換要求が発行された場合、置換すべきデータブロックをLRU等のデータブロック置換アルゴリズムにより、置換対象データブロックを検出する(ステップS40)。

【0057】当該置換対象データブロックがDIRTY状態の場合(ステップS41)、DIRTYブロックの主記憶へのライトバック処理を実行する(ステップS42)。当該置換対象データブロックがINVALID状態の場合や、当

該置換対象データブロックがCLEAN状態の場合、及び当該置換対象データブロックがDIRTY状態で主記憶へのライトバック処理が完了した場合、当該データブロックのデータエントリにデータブロックをLOADする(ステップS43、44)。

【0058】本手法の理解のために、まず本手法の特徴である共有データブロックに対するメモリ・アクセスに関しての一貫性保持動作が同期命令発行時点まで延期される例を説明する。

【0059】具体的には、例えばプロセッサ10、20が非共有データブロックであるアドレスf800000番地に対してSTORE命令を発行した場合、STORE命令発行時点、及びその後の同期命令や共有データブロックアクセス開始命令発行時点ではキャッシュユニット21に対して一貫性保持動作は発行されない点、また、共有データブロックに対してメモリ・アクセスするために共有データブロックアクセス開始命令を発行し、プロセッサ10が共有データブロックであるアドレスf8010000番地に対してSTORE命令を発行した場合、STORE命令発行時点ではキャッシュユニット21に対して一貫性保持動作は発生せず、その後、同期命令がプロセッサ10から発行された場合に、一貫性保持動作が実行される点がどのようにして実現するかを図10を参考に説明する。

【0060】図10は、本実施の形態の一貫性保持動作の一例を示すタイミングチャート図である。

【0061】アドレスf800000番地、及びアドレスf8010000番地が主記憶ユニット12に割り当てられているものとして説明する。

【0062】但し、時刻1以前には共有データブロックアクセス終了命令の実行が完了しているか、又は共有データブロックアクセス開始命令が実行されているものどちらかで、共有データブロック・アクセスフラグ118はリセットされた状態であり、かつ、アドレスf800000番地、及びアドレスf8010000番地のどちらもキャッシュユニット内にコピーをLOADしていないとする。

【0063】時刻1では、プロセッサ10、20が非共有データブロックとなるアドレスf800000番地に対してLOAD命令を発行し、それぞれ主記憶ユニット12からLOADを完了している。このとき、アドレスf800000番地に対応するキャッシュユニット11、21の内の状態フラグ113、213は、それぞれCLEANであり、かつ、共有データブロックフラグ114はリセットされている。

【0064】時刻2では、プロセッサ10が非共有データブロックとなるアドレスf800000番地に対してSTORE命令を発行し、STOREが完了している。このとき、

ローカルバスに対するアクセス、及び一貫性保持動作は発生しない。キャッシュ11内の当該データブロックの状態フラグ113はプロセッサ10の発行したSTORE命令の結果、DIRTYになっており、かつ、共有データブロック・アクセスフラグ118がリセットされているため、当該データブロックの共有データブロックフラグ114はリセットされる。

【0065】時刻3では、プロセッサ10が共有データブロックに対してメモリ・アクセスを開始するために、共有データブロックアクセス開始命令を発行し、共有データブロックアクセス開始命令が完了している。

【0066】時刻4では、プロセッサ10が共有データブロックとなるアドレスf8010000番地に対してSTORE命令を発行し、STORE命令が完了している。このとき、ローカルバスに対するアクセス、及び一貫性保持動作は発生しない。キャッシュ11内の当該データブロックの状態フラグ113はプロセッサ10の発行したSTORE命令の結果、DIRTYになっており、かつ、時刻2で完了した共有データブロックアクセス開始命令により、共有データブロック・アクセスフラグ118がセットされているため、共有データブロックフラグ114がセットされる。

【0067】時刻5では、プロセッサ10が共有データブロックに関するメモリ・アクセスを終了し、共有データブロックアクセス終了命令を発行し、共有データブロックアクセス終了命令が完了している。

【0068】時刻6では、プロセッサ10が同期命令を発行している。

【0069】時刻7では、時刻6で発行されたプロセッサ10の同期命令により、キャッシュ11内にDIRTYで保持されているデータブロックのうち、共有データブロックフラグ114がセットされている共有データブロックであるアドレスf8010000番地の主記憶へのライトバック処理が生じている。同じくキャッシュ11内にDIRTYで保持されているアドレスf8000000番地のデータブロックは、共有データブロックフラグ114がリセットされている非共有データブロックであるため上記処理のライトバック処理は生じない。

【0070】＜第2の実施形態＞次に、第2の実施形態について説明するが、システムの構成及び、キャッシュユニットの構成及び制御等については、上記第1の実施形態とほぼ同じであるので説明は省略し、異なる点のみについて説明する。

【0071】図1において、プロセッサ10は、メモリ・アクセスを多重発行することが可能であり、多重発行されたメモリ・アクセスを緩いメモリ・コンシステンシーモデルによるデータ一貫性保証の下で完了させるための特別な命令(同期命令)を持つものとするが、プロセッサ10は、共有データブロックに対するメモリ・アクセスを開始することを宣言する特別な命令(共有データブ

ロックアクセス開始命令)、及び共有データブロックに対するメモリ・アクセスを終了することを宣言する特別な命令(共有データブロックアクセス終了命令)を持たないものとする。

【0072】本第2の実施形態では、実施の形態1の共有データブロックアクセス開始命令、及び共有データブロックアクセス終了命令の代りに、共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを開始したことを示す値をSTOREすることにより、共有データブロックに対するメモリ・アクセスを開始することを宣言し、共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを終了したことを示す値をSTOREすることにより、共有データブロックに対するメモリ・アクセスを終了することを宣言するものとする。また、共有データブロックに対するメモリ・アクセスの開始・終了を共有データブロックアクセス開始アドレスにSTOREされている値で判断することにより、キャッシュユニット11は、共有データブロック・アクセスフラグ118を、プロセッサ10が共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを開始したことを示す値をSTOREするとセットし、共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを終了したことを示す値をSTOREするとリセットするものとする。

【0073】本実施形態では、プロセッサから明示的に与えられる同期命令発行時点でデータブロックの一貫性が保証されるような緩いメモリ・コンシステンシーモデルを採用した一貫性制御が行われる図1に示すような構成の情報処理システムにおいて、キャッシュの一貫性保持動作を制御するキャッシュ制御シーケンスが、共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを開始したことを示す値をSTOREした場合に、以後のメモリ・アクセスに対して共有データブロックアクセス開始アドレスに共有データブロックに対するメモリ・アクセスを終了したことを示す値をSTOREするまで、共有データブロック・アクセスフラグをセットし、上記値にその値を反映していない状態(以下DIRTY状態)にあるキャッシュのデータブロックを、共有データブロックであるかどうか判別し、同期命令発行時点で共有データブロックのみの主記憶へのライトバック処理、及び必要ならば一貫性保持動作を実行することによって、同期命令発行時点で集中的に発生する主記憶へのライトバック処理トラフィックや一貫性保持動作・トラフィックのうち非共有データブロックに対するトラフィックを発生させないことを実現している。

【0074】以上説明したように、本実施形態によれば、緩いメモリ・コンシステンシーモデルを採用し、相互結合網で互いに結合された並列計算機システムにおいて、その性能をより向上させるためのキャッシュ・メモ

り及びキャッシュ・メモリの一貫性保持動作機構を提供し、同期ポイント時点で集中する相互結合網上のトラフィックを共有データブロックに対するものだけに限定して発生させることで、相互結合網の利用効率低下、及び同期動作に伴う処理のオーバーヘッドを軽減することが可能であり、システム全体の処理能力を向上させるという効果がある。

【0075】尚、本願発明は複数のプロセッサ（上記実施形態では2個としたがそれ以上であっても勿論構わない）に適合することに特徴があるのであって、そのシステムあるいは装置は如何なる用途に用いられるものであっても良いし、汎用のコンピュータ装置であっても良いのは勿論である。

【0076】

【発明の効果】以上説明したように、本発明によれば、キャッシュの一貫性保持動作の実施を都合の良い時点まで遅延させ、共有データブロックに対してのみ一貫性保持を行うことで、例えば、緩いメモリ・コンシステンシ・モデルのシステムに、同期ポイント時点における相互結合網上のトラフィックの集中による相互結合網の利用効率の低下を避け、より性能の高いマルチプロセッサ装置が提供できる。

【0077】

【図面の簡単な説明】

【図1】本発明の実施形態における情報処理システムのプロセッサ周辺の回路構成を示す図である。

【図2】実施形態におけるキャッシュユニットの構成を示す図である。

【図3】実施形態のキャッシュユニットにおけるアドレス比較を説明する図である。

【図4】実施形態の状態フラグの状態遷移図である。

【図5】実施形態においてキャッシュユニットがLOAD命令実行時に行う処理のチャートである。

【図6】実施形態においてキャッシュユニットがSTORE命令実行時に行う処理のチャートである。

【図7】実施形態においてキャッシュユニットが同期命令実行時に行う処理のチャートである。

【図8】実施形態において主記憶へのライトバック処理実施時のチャートである。

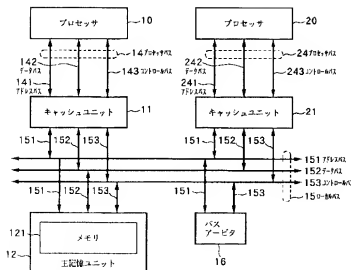
【図9】実施形態においてデータブロックの置換時に行う処理のチャート図である。

【図10】実施形態において同期命令発行時点まで共有データブロックに対するメモリ・アクセスに関しての一貫性保持動作が延期される例を示す図である。

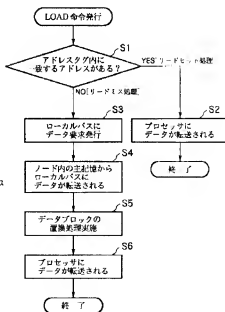
【符号の説明】

- 10、20 プロセッサ
- 11、21 キャッシュユニット
- 12 主記憶ユニット
- 14、24 プロセッサバス
- 16 バスアービタ
- 15 ローカルバス

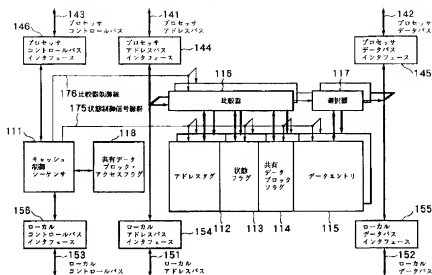
【図1】



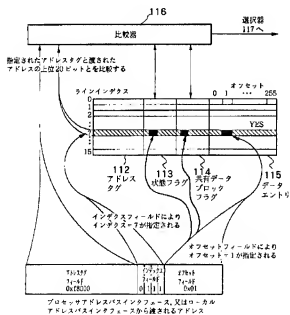
【図5】



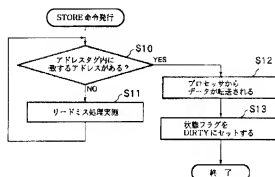
【図2】



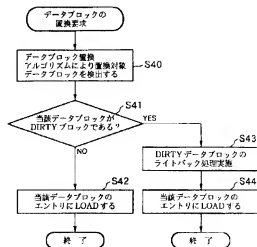
【図3】



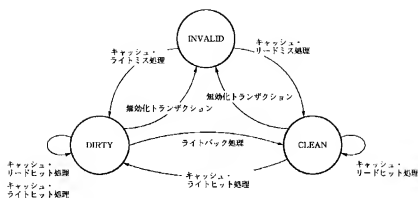
【図6】



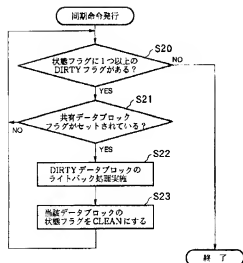
【図9】



【図1】



【図7】



【図8】

